1   Static Semantics

$\boxed{\Theta; \Delta; \Gamma \vdash e : t}$ Typing rules for expressions

$$\frac{}{\Theta; \Delta; \cdot, x : t \vdash x : t} \quad \text{Ty\_Var\_Lin}$$

$$\frac{x : t \in \Delta}{\Theta; \Delta; \cdot \vdash x : t} \quad \text{Ty\_Var}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \Gamma \vdash e : t \\ \Theta; \Delta; \Gamma', x : t \vdash e' : t' \end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\, x = e\, \mathbf{in}\, e' : t'} \quad \text{Ty\_Let}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash () : \mathbf{unit}} \quad \text{Ty\_Unit\_Intro}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \Gamma \vdash e : \mathbf{unit} \\ \Theta; \Delta; \Gamma' \vdash e' : t \end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\, () = e\, \mathbf{in}\, e' : t} \quad \text{Ty\_Unit\_Elim}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash \mathbf{true} : \mathbf{bool}} \quad \text{Ty\_Bool\_True}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash \mathbf{false} : \mathbf{bool}} \quad \text{Ty\_Bool\_False}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \Gamma \vdash e :\, !\mathbf{bool} \\ \Theta; \Delta; \Gamma' \vdash e_1 : t' \\ \Theta; \Delta; \Gamma' \vdash e_2 : t' \end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{if}\, e\, \mathbf{then}\, e_1\, \mathbf{else}\, e_2 : t} \quad \text{Ty\_Bool\_Elim}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash k : \mathbf{int}} \quad \text{Ty\_Int\_Intro}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash el : \mathbf{elt}} \quad \text{Ty\_Elt\_Intro}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \cdot \vdash v : t \\ v \neq l \end{array}}{\Theta; \Delta; \cdot \vdash \mathbf{Many}\, v :\, !t} \quad \text{Ty\_Bang\_Intro}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \Gamma \vdash e :\, !t \\ \Theta; \Delta, x : t; \Gamma' \vdash e' : t' \end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\, \mathbf{Many}\, x = e\, \mathbf{in}\, e' : t'} \quad \text{Ty\_Bang\_Elim}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \Gamma \vdash e : t \\ \Theta; \Delta; \Gamma' \vdash e' : t' \end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash (e, e') : t \otimes t'} \quad \text{Ty\_Pair\_Intro}$$

$$\frac{\begin{array}{c} \Theta; \Delta; \Gamma \vdash e_{12} : t_1 \otimes t_2 \\ \Theta; \Delta; \Gamma', a : t_1, b : t_2 \vdash e : t \end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\, (a, b) = e_{12}\, \mathbf{in}\, e : t} \quad \text{Ty\_Pair\_Elim}$$

$$\frac{\begin{array}{c}\Theta \vdash t' \text{ Type} \\ \Theta; \Delta; \Gamma, x : t' \vdash e : t\end{array}}{\Theta; \Delta; \Gamma \vdash \mathbf{fun}\, x : t' \to e : t' \multimap t} \quad \text{Ty\_Lambda}$$

$$\frac{\begin{array}{c}\Theta; \Delta; \Gamma \vdash e : t' \multimap t \\ \Theta; \Delta; \Gamma' \vdash e' : t'\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash e\, e' : t} \quad \text{Ty\_App}$$

$$\frac{\Theta, fc; \Delta; \Gamma \vdash e : t}{\Theta; \Delta; \Gamma \vdash \mathbf{fun}\, fc \to e : \forall fc.t} \quad \text{Ty\_Gen}$$

$$\frac{\begin{array}{c}\Theta \vdash f \text{ Perm} \\ \Theta; \Delta; \Gamma \vdash e : \forall fc.t\end{array}}{\Theta; \Delta; \Gamma \vdash e[f] : t[f/fc]} \quad \text{Ty\_Spc}$$

$$\frac{\Theta; \Delta, g : t \multimap t'; \cdot, x : t \vdash e : t'}{\Theta; \Delta; \cdot \vdash \mathbf{fix}\,(g, x : t, e : t') : t \multimap t'} \quad \text{Ty\_Fix}$$

2 Dynamic Semantics

$$\boxed{\langle \sigma, e \rangle \to Config} \quad \text{Operational semantics}$$

$$\frac{}{\langle \sigma, \mathbf{let}\,() = ()\,\mathbf{in}\,e \rangle \to \langle \sigma, e \rangle} \quad \text{Op\_Let\_Unit}$$

$$\frac{}{\langle \sigma, \mathbf{let}\,x = v\,\mathbf{in}\,e \rangle \to \langle \sigma, e[x/v] \rangle} \quad \text{Op\_Let\_Var}$$

$$\frac{}{\langle \sigma, \mathbf{if}\,(\mathbf{Many\ true})\,\mathbf{then}\,e_1\,\mathbf{else}\,e_2 \rangle \to \langle \sigma, e_1 \rangle} \quad \text{Op\_If\_True}$$

$$\frac{}{\langle \sigma, \mathbf{if}\,(\mathbf{Many\ false})\,\mathbf{then}\,e_1\,\mathbf{else}\,e_2 \rangle \to \langle \sigma, e_2 \rangle} \quad \text{Op\_If\_False}$$

$$\frac{}{\langle \sigma, \mathbf{let}\,\mathbf{Many}\,x = \mathbf{Many}\,v\,\mathbf{in}\,e \rangle \to \langle \sigma, e[x/v] \rangle} \quad \text{Op\_Let\_Many}$$

$$\frac{}{\langle \sigma, \mathbf{let}\,(a, b) = (v_1, v_2)\,\mathbf{in}\,e \rangle \to \langle \sigma, e[a/v_1][b/v_2] \rangle} \quad \text{Op\_Let\_Pair}$$

$$\frac{}{\langle \sigma, (\mathbf{fun}\,fc \to v)[f] \rangle \to \langle \sigma[fc/f], v[fc/f] \rangle} \quad \text{Op\_Frac\_Perm}$$

$$\frac{}{\langle \sigma, \mathbf{fix}\,(g, x : t, e : t')\,v \rangle \to \langle \sigma, e[x/v][g/\mathbf{fix}\,(g, x : t, e : t')] \rangle} \quad \text{Op\_App\_Fix}$$

$$\frac{}{\langle \sigma, (\mathbf{fun}\,x : t \to e)\,v \rangle \to \langle \sigma, e[x/v] \rangle} \quad \text{Op\_App\_Lambda}$$

$$\frac{\langle \sigma, e \rangle \to \langle \sigma', e' \rangle}{\langle \sigma, C[e] \rangle \to \langle \sigma', C[e'] \rangle} \quad \text{Op\_Context}$$

$$\frac{\langle\sigma,e\rangle\to\mathbf{err}}{\langle\sigma,C[e]\rangle\to\mathbf{err}}\quad\text{Op\_Context\_Err}$$

$$\frac{0\le k_1,k_2\quad l\text{ fresh}}{\langle\sigma,\mathbf{matrix}\,k_1\,k_2\rangle\to\langle\sigma+\{l\mapsto_1 M_{k_1,k_2}\},l\rangle}\quad\text{Op\_Matrix}$$

$$\frac{k_1<0\text{ or }k_2<0}{\langle\sigma,\mathbf{matrix}\,k_1\,k_2\rangle\to\mathbf{err}}\quad\text{Op\_Matrix\_Neg}$$

$$\frac{}{\langle\sigma+\{l\mapsto_1 m_{k_1,k_2}\},\mathbf{free}\,l\rangle\to\langle\sigma,()\rangle}\quad\text{Op\_Free}$$

$$\frac{}{\langle\sigma+\{l\mapsto_f m_{k_1,k_2}\},\mathbf{share}[f]\,l\rangle\to\langle\sigma+\{l\mapsto_{\frac12 f} m_{k_1,k_2}\}+\{l\mapsto_{\frac12 f} m_{k_1,k_2}\},(l,l)\rangle}\quad\text{Op\_Share}$$

$$\frac{}{\langle\sigma+\{l\mapsto_{\frac12 f} m_{k_1,k_2}\}+\{l\mapsto_{\frac12 f} m_{k_1,k_2}\},\mathbf{unshare}[f]\,l\,l\rangle\to\langle\sigma+\{l\mapsto_f m_{k_1,k_2}\},l\rangle}\quad\text{Op\_Unshare\_Eq}$$

$$\frac{l\ne l'}{\langle\sigma+\{l\mapsto_{\frac12 f} m_{k_1,k_2}\}+\{l'\mapsto_{\frac12 f} m'_{k_1,k_2}\},\mathbf{unshare}[f]\,l\,l'\rangle\to\mathbf{err}}\quad\text{Op\_Unshare\_Neq}$$

$$\frac{\begin{array}{l}\sigma'\equiv\sigma+\{l_1\mapsto_{fc_1} m_{1\,k_1,k_2}\}+\{l_2\mapsto_{fc_2} m_{2\,k_2,k_3}\}\\\sigma_1\equiv\sigma'+\{l_3\mapsto_1 m_{3\,k_1,k_3}\}\\\sigma_2\equiv\sigma'+\{l_3\mapsto_1 (m_1\,m_2+m_3)_{k_1,k_3}\}\end{array}}{\langle\sigma_1,\mathbf{gemm}[fc_1]\,l_1[fc_2]\,l_2\,l_3\rangle\to\langle\sigma_2,((l_1,l_2),l_3)\rangle}\quad\text{Op\_Gemm\_Match}$$

$$\frac{\begin{array}{l}k_2\ne k_2'\\\sigma'\equiv\sigma+\{l_1\mapsto_{fc_1} m_{1\,k_1,k_2}\}+\{l_2\mapsto_{fc_2} m_{2\,k_2',k_3}\}\end{array}}{\langle\sigma'+\{l_3\mapsto_1 m_{1\,k_1,k_3}\},\mathbf{gemm}[fc_1]\,l_1[fc_2]\,l_2\,l_3\rangle\to\mathbf{err}}\quad\text{Op\_Gemm\_Mismatch}$$

3 Interpretation

3.1 Definitions

Operationally, *Heap* $\sqsubseteq$ *Loc* $\times$ *Permission* $\times$ *Matrix* (a multiset), denoted with a σ.
Define its *interpretation* to be *Loc* $\rightharpoonup$ *Permission* $\times$ *Matrix* with $\star:$ *Heap* $\times$ *Heap* $\rightharpoonup$ *Heap* as follows:

$$(\varsigma_1\star\varsigma_2)(l)\equiv\begin{cases}\varsigma_1(l)&\text{if }l\in\operatorname{dom}(\varsigma_1)\wedge l\notin\operatorname{dom}(\varsigma_2)\\\varsigma_2(l)&\text{if }l\in\operatorname{dom}(\varsigma_2)\wedge l\notin\operatorname{dom}(\varsigma_1)\\(f_1+f_2,m)&\text{if }(f_1,m)=\varsigma_1(l)\wedge(f_2,m)=\varsigma_2(l)\wedge f_1+f_2\le 1\\\text{undefined}&\text{otherwise}\end{cases}$$

Commutativity and associativity of $\star$ follows from that of $+$.
$\varsigma_1\star\varsigma_2$ is *defined* if it is for all $l\in\operatorname{dom}(\varsigma_1)\cup\operatorname{dom}(\varsigma_2)$.
Define $\mathcal{H}[\![\sigma]\!]=\bigstar_{(l,f,m)\in\sigma}[l\mapsto_f m]$ and **implicitly denote** $\varsigma\equiv\mathcal{H}[\![\theta(\sigma)]\!]$.

The $n-$fold iteration for the $\to$ (functional) relation, is also a (functional) relation:

$$\forall n.\ \mathbf{err}\to^n\mathbf{err}\qquad\langle\sigma,v\rangle\to^n\langle\sigma,v\rangle\qquad\langle\sigma,e\rangle\to^0\langle\sigma,e\rangle\qquad\langle\sigma,e\rangle\to^{n+1}((\langle\sigma,e\rangle\to)\to^n)$$

Hence, all bounded iterations end in either an **err**, a heap-and-expression or a heap-and-value.

3

3.2 Interpretation

$$\mathcal{V}_k[\![\mathbf{unit}]\!] = \{(\emptyset, *)\}$$

$$\mathcal{V}_k[\![\mathbf{bool}]\!] = \{(\emptyset, true), (\emptyset, false)\}$$

$$\mathcal{V}_k[\![\mathbf{int}]\!] = \{(\emptyset, n) \mid 2^{-63} \le n \le 2^{63} - 1\}$$

$$\mathcal{V}_k[\![\mathbf{elt}]\!] = \{(\emptyset, f) \mid f \text{ a IEEE Float64 }\}$$

$$\mathcal{V}_k[\![f\,\mathbf{mat}]\!] = \{(\{l \mapsto_{2^{-f}} \_\}, l)\}$$

$$\mathcal{V}_k[\![!t]\!] = \{(\emptyset, \mathbf{Many}\ v) \mid (\emptyset, v) \in \mathcal{V}_k[\![t]\!]\}$$

$$\mathcal{V}_k[\![\forall fc.\ t]\!] = \{(\varsigma, \mathbf{fun}\ fc \to v) \mid \forall f.\ (\varsigma[fc/f], v[fc/f]) \in \mathcal{V}_{k-1}[\![t[fc/f]]\!]\}$$

$$\mathcal{V}_k[\![t_1 \otimes t_2]\!] = \{(\varsigma_1 \star \varsigma_2, (v_1, v_2)) \mid (\varsigma_1, v_1) \in \mathcal{V}_k[\![t_1]\!] \wedge (\varsigma_2, v_2) \in \mathcal{V}_k[\![t_2]\!]\}$$

$$\mathcal{V}_k[\![t' \multimap t]\!] = \{(\varsigma_v, v) \mid (v \equiv \mathbf{fun}\ x : t' \to e \vee v \equiv \mathbf{fix}(g, x : t', e : t)) \wedge$$
$$\forall j \le k, (\varsigma_{v'}, v') \in \mathcal{V}_j[\![t']\!].\ \varsigma_v \star \varsigma_v' \text{ defined } \Rightarrow (\varsigma_v \star \varsigma_v', v\,v') \in \mathcal{C}_j[\![t]\!]\}$$

$$\mathcal{C}_k[\![t]\!] = \{(\varsigma_s, e_s) \mid \forall j < k, \sigma_r.\ \varsigma_s \star \varsigma_r \text{ defined } \Rightarrow \langle \sigma_s + \sigma_r, e_s \rangle \to^j \mathbf{err}\ \vee \exists \sigma_f, e_f.$$
$$\langle \sigma_s + \sigma_r, e_s \rangle \to^j \langle \sigma_f + \sigma_r, e_f \rangle \wedge (e_f \text{ is a value } \Rightarrow (\varsigma_f \star \varsigma_r, e_f) \in \mathcal{V}_{k-j}[\![t]\!])\}$$

$$\mathcal{I}_k[\![\cdot]\!]\theta = \{[]\}$$

$$\mathcal{I}_k[\![\Delta, x : t]\!]\theta = \{\delta[x \mapsto v_x] \mid \delta \in \mathcal{I}_k[\![\Delta]\!]\theta \wedge (\emptyset, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]\}$$

$$\mathcal{L}_k[\![\cdot]\!]\theta = \{(\emptyset, [])\}$$

$$\mathcal{L}_k[\![\Gamma, x : t]\!]\theta = \{(\varsigma \star \varsigma_x, \gamma[x \mapsto v_x]) \mid (\varsigma, \gamma) \in \mathcal{L}_k[\![\Gamma]\!]\theta \wedge (\varsigma_x, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]\}$$

$$\mathcal{H}[\![\sigma]\!] = \bigstar_{(l,f,m)\in\sigma}[l \mapsto_f m]$$
$$\varsigma \equiv \mathcal{H}[\![\theta(\sigma)]\!]$$

$$_k[\![\Theta; \Delta; \Gamma \vdash e : t]\!] = \forall \theta, \delta, \gamma, \sigma.\ \Theta = \mathrm{dom}(\theta) \wedge (\varsigma, \gamma) \in \mathcal{L}_k[\![\Gamma]\!]\theta \wedge \delta \in \mathcal{I}_k[\![\Delta]\!]\theta \Rightarrow$$
$$(\varsigma, \theta(\delta(\gamma(e)))) \in \mathcal{C}_k[\![\theta(t)]\!]$$

4 Lemmas

4.1 $\forall \sigma_s, \sigma_r, e. \; \varsigma_s \star \varsigma_r$ **defined** $\Rightarrow \forall n. \; \langle \sigma_s, e \rangle \rightarrow^n = \langle \sigma_s + \sigma_r, e \rangle \rightarrow^n$

SUFFICES: By induction on n, consider only the cases $\langle \sigma_s, e \rangle \rightarrow \langle \sigma_f, e_f \rangle$ where $\sigma_s \neq \sigma_f$.

PROOF SKETCH: Only OP-{FREE, MATRIX, SHARE, UNSHARE_EQ, GEMM_MATCH} change the heap: the rest are either parametric in the heap or step to an **err**.

PROVE: $\langle \sigma_s + \sigma_r, e \rangle \rightarrow \langle \sigma_f + \sigma_r, e_f \rangle$.

$\langle 1 \rangle 1.$ CASE: OP_FREE, $\sigma_s \equiv \sigma' + \{l \mapsto_1 m\}$, $\sigma_f = \sigma'$.
PROOF: Instantiate OP_FREE with $(\sigma' + \sigma_r) + \{l \mapsto_1 m\}$,
valid because $l \notin \mathrm{dom}(\varsigma_r)$ by $\varsigma' \star [l \mapsto_1 m] \star \varsigma_r$ defined (assumption).

$\langle 1 \rangle 2.$ CASE: OP_MATRIX
PROOF: Rule has no requirements on σ_s so will also work with $\sigma_s + \sigma_r$.

$\langle 1 \rangle 3.$ CASE: OP_SHARE, $\sigma_s \equiv \sigma' + \{l \mapsto_f m\}$, $\sigma_f = \sigma' + \{l \mapsto_{\frac{1}{2} \cdot f} m\} + \{l \mapsto_{\frac{1}{2} \cdot f} m\}$.
PROOF: Union-ing σ_r does not remove $l \mapsto_f m$, so that can be split out of $\sigma_s + \sigma_r$ as before.

$\langle 1 \rangle 4.$ CASE: OP_UNSHARE_EQ, $\sigma_s \equiv \sigma' + \{l \mapsto_{\frac{1}{2} \cdot f} m\} + \{l \mapsto_{\frac{1}{2} \cdot f} m\}$, $\sigma_f = \sigma' + \{l \mapsto_f m\}$.

$\langle 2 \rangle 1.$ Union-ing σ_r does not remove $l \mapsto_{\frac{1}{2} \cdot f} m$, so that can still be split out of $\sigma_s + \sigma_r$.

$\langle 2 \rangle 2.$ There may also be other valid splits introduced by σ_r.

$\langle 2 \rangle 3.$ However, by assumption of $\varsigma_s \star \varsigma_r$ defined, any splitting of $\sigma_s + \sigma_r$ will satisfy $f \leq 1$.

$\langle 1 \rangle 5.$ CASE: OP_GEMM_MATCH

$\langle 2 \rangle 1.$ By assumption of $\varsigma_s \star \varsigma_r$ defined, either l_1 (or l_2, or both) are not in σ_r, or they are and the matrix values they point to are the same.

$\langle 2 \rangle 2.$ The permissions (of l_1 and/or l_2) may differ, but OP_GEMM_MATCH universally quantifies over them and leaves them unchanged, so they are irrelevant.

$\langle 2 \rangle 3.$ Only the pointed to matrix value at l_3 changes.

$\langle 2 \rangle 4.$ SUFFICES: $l_3 \notin \pi_1[\sigma_r]$.

$\langle 2 \rangle 5.$ By assumption of $\varsigma_s \star \varsigma_r$ defined, $l_3 \notin \mathrm{dom}(\varsigma_r)$.

$\langle 2 \rangle 6.$ Hence $l_3 \notin \pi_1[\sigma_r]$.

4.2 $\forall k, t. \; \mathcal{V}_k[\![t]\!] \subseteq \mathcal{C}_k[\![t]\!]$

Follows from definition of $\mathcal{C}_k[\![t]\!]$, $\rightarrow^j$ $(\forall n. \; \langle \sigma, v \rangle \rightarrow^n \langle \sigma, v \rangle)$ for arbitrary $j \leq k$ and 4.1.

4.3 $\forall \theta, \delta, \gamma, v. \; \theta(\delta(\gamma(v)))$ **is a value.**

θ is irrelevant because it only maps fractional permission variables to fractional permissions. By construction, δ and γ only map variables to values, and values are closed under substitution.

4.4 $\forall k, \sigma, \sigma', e, e', t. \ (\varsigma', e') \in \mathcal{C}_k[\![t]\!] \land \langle \sigma, e \rangle \to \langle \sigma', e' \rangle \Rightarrow (\varsigma, e) \in \mathcal{C}_{k+1}[\![t]\!]$

In the lemma, and for the rest of its proof, $\varsigma = \mathcal{H}[\![\sigma]\!]$.

ASSUME: arbitrary $j < k + 1$, and σ_r such that $\varsigma \star \varsigma_r$ defined.

$\langle 1 \rangle 1.$ CASE: $j = 0$. Clearly $\sigma_f = \sigma_s + \sigma_r$ and $e' = e$.
Remains to show that if e is a value then $(\varsigma_s \star \varsigma_r, e) \in \mathcal{V}_k[\![t]\!]$.
This is true vacuously, because by assumption, e is not a value.

$\langle 1 \rangle 2.$ CASE: $j \geq 1$. We have $\langle \sigma, e \rangle \to^j = \langle \sigma', e' \rangle \to^{j-1}$.
Instantiate $(\varsigma', e') \in \mathcal{C}_k[\![t]\!]$, with $j - 1 < k$ and σ_r to conclude the required conditions.

4.5 $j \leq k \Rightarrow \_{\ k}[\![\cdot]\!] \subseteq \_{\ j}[\![\cdot]\!]$

For the rest of this proof, $\varsigma = \mathcal{H}[\![\sigma]\!]$.
Lemma 4.4 is the inductive step for this lemma for the $\mathcal{C}[\![]\!]$ case.
Need to prove for $\mathcal{V}[\![]\!]$, by induction on t and then index.

SUFFICES: Consider only $t \multimap t'$ case, rest use k directly on structure of type.
ASSUME: Arbitrary $j \leq k$ and $(\varsigma_{v'}, v') \in \mathcal{V}_k[\![t \multimap t']\!]$.
PROVE: $(\varsigma_{v'}, v') \in \mathcal{V}_j[\![t \multimap t']\!]$.

$\langle 1 \rangle 1.$ v' is of the correct syntactic form (lambda or fixpoint) by assumption.

$\langle 1 \rangle 2.$ ASSUME: arbitrary $j' \leq j$ and $(\varsigma_v, v) \in \mathcal{V}_{j'}[\![t]\!]$ such that $\varsigma_{v'} \star \varsigma_v$ is defined.

$\langle 1 \rangle 3.$ SUFFICES: to show $(\varsigma_{v'} \star \varsigma_v, v'v) \in \mathcal{C}_{j'}[\![t']\!]$.

$\langle 1 \rangle 4.$ This is true by instantiating $(\varsigma_{v'}, v') \in \mathcal{V}_k[\![t \multimap t']\!]$ with $j' \leq k$ and $(\varsigma_v, v) \in \mathcal{V}_{j'}[\![t]\!]$.

4.6 $\forall \Delta, \Gamma, t, k, \theta, \delta, \gamma. \ \delta \in \mathcal{I}_k[\![\Delta]\!]\theta \land \gamma \in \pi_2[\mathcal{L}_k[\![\Gamma]\!]\theta] \Rightarrow$
$\text{dom}(\Delta) = \text{dom}(\delta) \ \textbf{and} \ \text{dom}(\Gamma) = \text{dom}(\gamma)$

PROOF: By induction on Δ and Γ.

4.7 $\forall k, \Gamma, \Gamma', \theta, \sigma_+, \gamma_+. \ (\varsigma_+, \gamma) \in \mathcal{L}_k[\![\Gamma, \Gamma']\!]\theta \land \Gamma, \Gamma' \ \textbf{disjoint} \ \Rightarrow$
$\exists \sigma, \gamma, \sigma', \gamma'. \ \sigma_+ = \sigma + \sigma' \land \gamma, \gamma' \ \textbf{disjoint} \ \land \gamma_+ = \gamma \cup \gamma'$
$\land \ (\varsigma, \gamma) \in \mathcal{L}_k[\![\Gamma]\!] \land (\varsigma', \gamma') \in \mathcal{L}_k[\![\Gamma']\!]$

PROOF: By induction on Γ'.

4.8 $\forall e, \sigma, e', \sigma', \theta. \ \langle \sigma, e \rangle \to \langle \sigma', e' \rangle \Rightarrow \langle \theta(\sigma), \theta(e) \rangle \to \langle \theta(\sigma'), \theta(e') \rangle$

PROOF: By induction on $\to$.

$\langle 1 \rangle 1.$ ASSUME: Arbitrary $e, \sigma, e', \sigma', \theta$ such that $\langle \sigma, e \rangle \to \langle \sigma', e' \rangle$.

$\langle 1 \rangle 2.$ SUFFICES: To consider only the following rules which mention fractional permission variables.
OP_FRAC_PERM, OP_SHARE, OP_UNSHARE_(N)EQ and OP_GEMM_(MIS)MATCH.

$\langle 1 \rangle 3.$ CASE: OP_FRAC_PERM.

Because substitution avoids capture,
$$\langle\theta(\sigma),\theta((\mathbf{fun}\,fc \to v)\,[f])\rangle \to \langle\theta(\sigma'\,[fc/f]),\theta(v\,[fc/f])\rangle.$$

$\langle 1\rangle 4.$ The rest of the cases are parametric in their use of fractional permission variables and so will take the same step ater any substitution.

$\langle 1\rangle 5.$ COROLLARY: If $\langle\sigma\,[fc/f_1], e\,[fc/f_1]\rangle \to^n \langle\sigma_2, e_2'\rangle$ and $\langle\sigma\,[fc/f_2], e\,[fc/f_2]\rangle \to^n \langle\sigma_2, e_2'\rangle$, then $\exists\,\sigma, e'.\ \sigma_1 = \sigma\,[fc/f_1] \wedge \sigma_2 = \sigma\,[fc/f_2] \wedge e_1' = e'\,[fc/f_1] \wedge e_2' = e'\,[fc/f_2].$

5 Soundness

$$\forall \Theta, \Delta, \Gamma, e, t. \ \Theta; \Delta; \Gamma \vdash e : t \Rightarrow \forall k. \ _k[\![\Theta; \Delta; \Gamma \vdash e : t]\!]$$

PROOF SKETCH: Induction over the typing judgements.

ASSUME: 1. Arbitrary $\Theta, \Delta, \Gamma, e, t$ such that $\Theta; \Delta; \Gamma \vdash e : t$.
        2. Arbitrary $k, \theta, \delta, \gamma, \sigma$ such that:
          a. $\Theta = \mathrm{dom}(\theta)$
          b. $\delta \in \mathcal{I}_k[\![\Delta]\!]\theta$.
          c. $(\varsigma, \gamma) \in \mathcal{L}_k[\![\Gamma]\!]\theta$
        3. W.l.o.g., all variables are distinct, hence Θ, $\mathrm{dom}(\Delta)$ and $\mathrm{dom}(\Gamma)$ are disjoint so order of θ, δ and γ (as substitutions defined recursively over expressions) is irrelevant.

PROVE: $(\varsigma, \theta(\delta(\gamma(e)))) \in \mathcal{C}_k[\![\theta(t)]\!]$.
ASSUME: Arbitrary $j < k$ and σ_r, such that $\varsigma \star \varsigma_r$ defined.
SUFFICES: $\langle \sigma + \sigma_r, e \rangle \rightarrow^j \mathbf{err} \ \lor \exists \sigma_f, e_f. \ \langle \sigma + \sigma_r, e \rangle \rightarrow^j \langle \sigma_f + \sigma_r, e_f \rangle$
        $\land \ (e_f \text{ is a value} \ \Rightarrow (\varsigma_f \star \varsigma_r, e_f) \in \mathcal{V}_{k-j}[\![t]\!])$.
SUFFICES: By 4.1, to show $\langle \sigma, e \rangle \rightarrow^j \mathbf{err} \ \lor \exists \sigma_f, e_f. \ \langle \sigma, e \rangle \rightarrow^j \langle \sigma_f, e_f \rangle$
        $\land \ (e_f \text{ is a value} \ \Rightarrow (\varsigma_f, e_f) \in \mathcal{V}_{k-j}[\![t]\!])$

$\langle 1 \rangle 1.$ CASE: TY_LET.

    $\langle 2 \rangle 1.$ By induction,
        1. $\forall k. \ _k[\![\Theta; \Delta; \Gamma \vdash e : t]\!]$
        2. $\forall k. \ _k[\![\Theta; \Delta; \Gamma', x : t \vdash e' : t']\!]$.

    $\langle 2 \rangle 2.$ By 2c, 3 and 4.7, we know there exists the following (for all k):
        1. $(\varsigma_e, \gamma_e) \in \mathcal{L}_k[\![\Gamma]\!]$
        2. $\gamma = \gamma_e \cup \gamma_{e'}$
        3. $\sigma = \sigma_e + \sigma_{e'}$.

    $\langle 2 \rangle 3.$ So, using $k, \theta, \delta, \gamma_e, \sigma_e$, we have $(\varsigma_e, \theta(\delta(\gamma_e(e)))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

    $\langle 2 \rangle 4.$ By $\langle 2 \rangle 2$ $(\gamma = \gamma_e \cup \gamma_{e'})$, have $(\varsigma_e, \theta(\delta(\gamma(e)))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

    $\langle 2 \rangle 5.$ By definition of $\mathcal{C}_k[\![\cdot]\!]$ and $\langle 2 \rangle 2$, we instantiate with j and $\sigma_r = \sigma_{e'}$ to conclude that $\langle \theta(\sigma), \theta(\delta(\gamma(e))) \rangle$ either takes j steps to $\mathbf{err}$ or another heap-and-expression $\langle \sigma_f, e_f \rangle$.

    $\langle 2 \rangle 6.$ CASE: j steps to $\mathbf{err}$
        By OP_CONTEXT_ERR, the whole expression reduces to $\mathbf{err}$ in $j < k$ steps.

    $\langle 2 \rangle 7.$ CASE: j steps to another heap-and-expression.
        If it is not a value, then OP_CONTEXT runs j times and we are done.

    $\langle 2 \rangle 8.$ If it is, then $\exists i \leq j. \ (\varsigma_f, v_1) \in \mathcal{V}_{k-i}[\![\theta(t_1)]\!] \subseteq \mathcal{V}_{k-j}[\![\theta(t_1)]\!]$ by 4.3 and 4.5.
        So, OP_CONTEXT runs i times, and then we have the following.
        SUFFICES: $(\varsigma_f \star \varsigma_{e'}, \mathbf{let} \ x = v \ \mathbf{in} \ \theta(\delta(\gamma(e')))) \in \mathcal{C}_{k-i}[\![\theta(t')]\!]$ by 4.4 i times.
        SUFFICES: $(\varsigma_f \star \varsigma_{e'}, \theta(\delta(\gamma(e')))[x/v]) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$ by 4.4.

    $\langle 2 \rangle 9.$ By 4.5, $(\varsigma_{e'}, \gamma_{e'}[x \mapsto v]) \in \mathcal{L}_k[\![\Gamma', x : t]\!]\theta \subseteq \mathcal{L}_{k-i-1}[\![\Gamma', x : t]\!]\theta$.

    $\langle 2 \rangle 10.$ Instantiate 2 of step $\langle 2 \rangle 1$ with $k - i - 1, \theta, \delta, \gamma_{e'}[x \mapsto v], \sigma_{e'}$ to conclude
        $(\varsigma_{e'}, \theta(\delta(\gamma_{e'}[x \mapsto v](e')))) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$.

$\langle 2 \rangle 11$. By 3, we have $\theta(\delta(\gamma(e')))[x/v] = \theta(\delta(\gamma_{e'}[x \mapsto v](e')))$ and
by 4.1 we conclude $(\varsigma_f \star \varsigma_{e'}, \theta(\delta(\gamma(e')))[x/v]) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$

$\langle 1 \rangle 2$. CASE: TY_PAIR_ELIM.
PROOF SKETCH: Similar to TY_LET, but with the following key differences.

$\langle 2 \rangle 1$. When $(\varsigma_f, v) \in \mathcal{V}_{k-i}[\![\theta(t_1) \otimes \theta(t_2)]\!]$, we have $v = (v_1, v_2)$.

$\langle 2 \rangle 2$. SUFFICES: $(\varsigma_{e'}, \theta(\delta(\gamma(e')))) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$ by 4.4 $i+1$ times.

$\langle 2 \rangle 3$. By 4.5, $(\varsigma_{e'}, \gamma_{e'}[a \mapsto v_1, b \mapsto v_2]) \in \mathcal{L}_k[\![\Gamma', a : t_1, b : t_2]\!]\theta \subseteq \mathcal{L}_{k-i-1}[\![\Gamma', a : t_1, b : t_2]\!]\theta$.

$\langle 2 \rangle 4$. Instantiate $_{k-i-1}[\![\Theta; \Delta; \Gamma', a : t_1, b : t_2 \vdash e' : t']\!]$ with $\theta, \delta, \gamma_{e'}[a \mapsto v_1, b \mapsto v_2], \sigma_{e'}$.

$\langle 2 \rangle 5$. By 3 (for $\gamma = \gamma_e \cup \gamma_{e'}$ and a, b), conclude $(\varsigma_{e'}, \theta(\delta(\gamma(e'[a/v_1][b/v_2])))) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$.

$\langle 1 \rangle 3$. CASE: TY_BANG_ELIM.
PROOF SKETCH: Similar to TY_LET, but with the following key differences.

$\langle 2 \rangle 1$. When $(\varsigma_f, v) \in \mathcal{V}_{k-i}[\![\theta(!t)]\!]$, since $\mathcal{V}_{k-i}[\![\theta(!t)]\!] = \mathcal{V}_{k-i}[\![!\theta(t)]\!]$,
we have $\varsigma_f = \emptyset$ and $v = \mathbf{Many}\, v'$ for some $(\emptyset, v') \in \mathcal{V}_{k-i}[\![\theta(t)]\!]$.

$\langle 2 \rangle 2$. SUFFICES: $(\varsigma_{e'}, \mathbf{let\, Many}\, x = \mathbf{Many}\, v'\, \mathbf{in}\, \theta(\delta(\gamma(e')))) \in \mathcal{C}_{k-i}[\![\theta(t)]\!]$.

$\langle 2 \rangle 3$. SUFFICES: $(\varsigma_{e'}, \theta(\delta(\gamma(e')))[x/v]) \in \mathcal{C}_{k-i-1}[\![\theta(t)]\!]$ by 4.4 $i+1$ times.

$\langle 2 \rangle 4$. Instantiate $_{k-i-1}[\![\Theta; \Delta, x : t, \Gamma' \vdash e' : t']\!]$ with $\theta, \delta_{e'} = \delta[x \mapsto v'], \gamma_{e'}, \sigma_{e'}$.

$\langle 2 \rangle 5$. By 3, $(\varsigma_{e'}, \theta(\delta(\gamma(e')))[x/v]) \in \mathcal{C}_{k-i-1}[\![\theta(t)]\!]$.

$\langle 1 \rangle 4$. CASE: TY_UNIT_ELIM.
PROOF SKETCH: Similar to TY_LET, but with the following key differences.

$\langle 2 \rangle 1$. When $(\varsigma_f, v) \in \mathcal{V}_{k-i}[\![\mathbf{unit}]\!]$, we have $\varsigma_f = \emptyset$ and $v = ()$.

$\langle 2 \rangle 2$. SUFFICES: $(\varsigma_{e'}, \theta(\delta(\gamma(e')))) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$ by 4.4 $i+1$ times.

$\langle 2 \rangle 3$. By 4.5, $(\varsigma_{e'}, \gamma_{e'}) \in \mathcal{L}_k[\![\Gamma']\!]\theta \subseteq \mathcal{L}_{k-i-1}[\![\Gamma']\!]\theta$.

$\langle 2 \rangle 4$. Instantiate $_{k-i-1}[\![\Theta; \Delta; \Gamma' \vdash e' : t']\!]$ with $\theta, \delta, \gamma_{e'}, \sigma_{e'}$.

$\langle 2 \rangle 5$. By 3 $(\varsigma_{e'}, \theta(\delta(\gamma(e')))) \in \mathcal{C}_{k-i-1}[\![\theta(t')]\!]$.

$\langle 1 \rangle 5$. CASE: TY_BOOL_ELIM.
PROOF SKETCH: Similar to TY_UNIT_ELIM but with OP_IF_{TRUE,FALSE}, $\varsigma_f = \emptyset$ and
$v = \mathbf{Many\, true}$ or $v = \mathbf{Many\, false}$.

$\langle 1 \rangle 6$. CASE: TY_BANG_INTRO.

$\langle 2 \rangle 1$. We have, $e = v$ for some value $v \neq l$, $\Gamma = \emptyset$ and so
$\forall k.\, _k[\![\Theta; \Delta; \cdot \vdash v : t]\!]$ by induction.

$\langle 2 \rangle 2$. SUFFICES: $(\emptyset, \mathbf{Many}\, \theta(\delta(v))) \in \mathcal{C}_k[\![!\theta(t)]\!]$ by 2c $(\varsigma = \emptyset, \gamma = [])$.

$\langle 2 \rangle 3$. Instantiate $_k[\![\Theta; \Delta; \cdot \vdash v : t]\!]$ with $\theta, \delta, \gamma = [], \sigma = \emptyset$ to obtain $(\emptyset, \theta(\delta(v))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

$\langle 2 \rangle 4$. Instantiate $(\emptyset, \theta(\delta(v))) \in \mathcal{C}_k[\![\theta(t)]\!]$ with $j = 0$, $\sigma_r = \emptyset$ and 4.3 $(\theta(\delta(v))$ is a value),

to conclude $(\emptyset, \theta(\delta(v))) \in \mathcal{V}_k[\![\theta(t)]\!]$.

$\langle 2 \rangle 5.$ By definition of $\mathcal{V}_k[\![!\theta(t)]\!]$, 4.3 and 4.2 we have $(\emptyset, \mathbf{Many}\ \theta(\delta(v))) \in \mathcal{C}_k[\![!\theta(t)]\!]$.

$\langle 1 \rangle 7.$ CASE: TY_PAIR_INTRO.

$\langle 2 \rangle 1.$ By 2c, 3 and 4.7, we know there exists the following (for all k):
1. $(\varsigma_1, \gamma_1) \in \mathcal{L}_k[\![\Gamma_1]\!]$
2. $(\varsigma_2, \gamma_2) \in \mathcal{L}_k[\![\Gamma_2]\!]$
3. $\gamma = \gamma_1 \cup \gamma_2$
4. $\sigma = \sigma_1 + \sigma_2.$

$\langle 2 \rangle 2.$ By induction,
1. $\forall k.\ {}_k[\![\Theta; \Delta; \Gamma_1 \vdash e_1 : t_1]\!]$
2. $\forall k.\ {}_k[\![\Theta; \Delta; \Gamma_2 \vdash e_2 : t_2]\!].$

$\langle 2 \rangle 3.$ Instantiate the first with $k, \theta, \delta, \gamma_1, \sigma_1$.

$\langle 2 \rangle 4.$ By that and $\langle 2 \rangle 1$, $(\varsigma_1, \theta(\delta(\gamma_1(e_1)))) = (\varsigma_1, \theta(\delta(\gamma(e_1)))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

$\langle 2 \rangle 5.$ So, $\langle \theta(\sigma_1 + \sigma_2), \theta(\delta(\gamma_1(e_1))) \rangle$ either takes j steps to **err** or a heap-and-expression $\langle \sigma_{1f}, e_{1f} \rangle$.

$\langle 2 \rangle 6.$ CASE: j steps to **err**
By OP_CONTEXT_ERR, the whole expression reduces to **err** in $j < k$ steps.

$\langle 2 \rangle 7.$ CASE: j steps to another heap-and-expression.
If it is not a value, then OP_CONTEXT runs j times and we are done.

$\langle 2 \rangle 8.$ If it is, then $\exists i_1 \leq j.\ (\varsigma_{1f}, v_1) \in \mathcal{V}_{k-i_1}[\![\theta(t_1)]\!] \subseteq \mathcal{V}_{k-j}[\![\theta(t_1)]\!]$ by 4.3 and 4.5.
So, OP_CONTEXT runs i_1 times, and then we have the following.
SUFFICES: By 4.4, $(\varsigma_{1f} \star \varsigma_2, (v_1, e_2)) \in \mathcal{C}_{k-i_1}[\![\theta(t_1 \otimes t_2)]\!]$.

$\langle 2 \rangle 9.$ Instantiate the second IH with $k, \theta, \delta, \gamma_2, \sigma_2$.

$\langle 2 \rangle 10.$ So, $\langle \theta(\sigma_{1f} + \sigma_2), \theta(\delta(\gamma_2(e_2))) \rangle$ either takes j steps to **err** or a heap-and-expression $\langle \sigma_{2f}, e_{2f} \rangle$.

$\langle 2 \rangle 11.$ CASE: j steps to **err**
By OP_CONTEXT_ERR, the whole expression reduces to **err** in $j < k$ steps.

$\langle 2 \rangle 12.$ CASE: j steps to another heap-and-expression.
If it is not a value, then OP_CONTEXT runs j times and we are done.

$\langle 2 \rangle 13.$ If it is, then $\exists i_2 \leq j.\ (\varsigma_{2f}, v_2) \in \mathcal{V}_{k-i_2}[\![\theta(t_2)]\!] \subseteq \mathcal{V}_{k-j}[\![\theta(t_2)]\!]$ by 4.3 and 4.5.
So, OP_CONTEXT runs i_2 times, and then we have the following.
SUFFICES: By 4.4, $(\varsigma_{1f} \star \varsigma_{2f}, (v_1, v_2)) \in \mathcal{V}_{k-i_1-i_2}[\![\theta(t_1) \otimes \theta(t_2)]\!]$.

$\langle 2 \rangle 14.$ By 4.5 and $k - i_1 - i_2 \leq k - i_1, k - i_2$, have
$(\varsigma_{1f}, v_1) \in \mathcal{V}_{k-i_1}[\![\theta(t_1)]\!] \subseteq \mathcal{V}_{k-i_1-i_2}[\![\theta(t_1)]\!]$ and
$(\varsigma_{2f}, v_2) \in \mathcal{V}_{k-i_2}[\![\theta(t_2)]\!] \subseteq \mathcal{V}_{k-i_1-i_2}[\![\theta(t_2)]\!]$ as needed.

$\langle 1 \rangle 8.$ CASE: TY_LAMBDA.
SUFFICES: By 4.2, to show $(\varsigma, \theta(\delta(\gamma(\mathbf{fun}\, x : t \to e)))) \in \mathcal{V}_k[\![\theta(t \multimap t')]\!]$.
ASSUME: Arbitrary $j \leq k$, $(\varsigma_v, v) \in \mathcal{V}_j[\![\theta(t)]\!]$ such that $\varsigma \star \varsigma_v$ is defined.
SUFFICES: $(\varsigma \star \varsigma_v, \theta(\delta(\gamma(\mathbf{fun}\, x : t \to e)))\, v) \in \mathcal{C}_j[\![\theta(t')]\!]$.

Suffices: $(\varsigma \star \varsigma_v, \theta(\delta(\gamma(e)))[x/v]) \in \mathcal{C}_{j-1}[\![\theta(t')]\!]$ by 4.4.

⟨2⟩1. By induction, $\forall k.\ _k[\![\Theta; \Delta; \Gamma, x : t \vdash e]\!]$.

⟨2⟩2. Instantiate it $j - 1, \theta, \delta, \gamma[x \mapsto v], \sigma + \sigma_v$.

⟨2⟩3. Hence, $(\varsigma \star \varsigma_v, \theta(\delta(\gamma[x \mapsto v](e)))) \in \mathcal{C}_{j-1}[\![\theta(t')]\!]$.

⟨2⟩4. By 3, $\theta(\delta(\gamma[x \mapsto v](e))) = \theta(\delta(\gamma(e)))[x/v]$, we are done.

⟨1⟩9. CASE: TY_APP.

⟨2⟩1. By 2c, 3 and 4.7, we know there exists the following (for all k):
1. $(\varsigma_e, \gamma_e) \in \mathcal{L}_k[\![\Gamma_e]\!]$
2. $(\varsigma_{e'}, \gamma_{e'}) \in \mathcal{L}_k[\![\Gamma_{e'}]\!]$
3. $\gamma = \gamma_e \cup \gamma_{e'}$
4. $\sigma = \sigma_e + \sigma_{e'}$.

⟨2⟩2. By induction,
1. $\forall k.\ _k[\![\Theta; \Delta; \Gamma \vdash e : t' \multimap t]\!]$
2. $\forall k.\ _k[\![\Theta; \Delta; \Gamma' \vdash e' : t']\!]$.

⟨2⟩3. Instantiate the first with $k, \theta, \delta, \gamma_e, \sigma_e$ to conclude $(\varsigma_e, \theta(\delta(\gamma_e(e)))) \in \mathcal{C}_k[\![\theta(t') \multimap \theta(t)]\!]$.

⟨2⟩4. Instantiate *this* with j and $\sigma_{e'}$ and use ⟨2⟩1 to conclude $\langle \theta(\sigma_e + \sigma_{e'}), \theta(\delta(\gamma(e))) \rangle$ either takes j steps to **err** or a heap-and-expression $\langle \sigma_f + \sigma_{e'}, e_f \rangle$.

⟨2⟩5. CASE: j steps to **err**
By OP_CONTEXT_ERR, the whole expression reduces to **err** in $j < k$ steps.

⟨2⟩6. CASE: j steps to another heap-and-expression.
If it is not a value, then OP_CONTEXT runs j times and we are done.

⟨2⟩7. If it is, then $\exists i_e \leq j.\ (\varsigma_f, e_f) \in \mathcal{V}_{k-i_e}[\![\theta(t') \multimap \theta(t)]\!] \subseteq \mathcal{V}_{k-j}[\![\ldots]\!]$ by 4.3 and 4.5.
So, OP_CONTEXT runs i_e times, and then we have the following.
Suffices: By 4.4 i_e times, $(\varsigma_f \star \varsigma_{e'}, e_f\, e') \in \mathcal{C}_{k-i_e}[\![\theta(t')]\!]$.

⟨2⟩8. By 4.5, $(\varsigma_{e'}, \gamma_{e'} \in \mathcal{L}_k[\![\Gamma']\!]\theta \subseteq \mathcal{L}_{k-i_e}[\![\Gamma']\!]\theta$.

⟨2⟩9. So, instantiate the second IH with $k - i_e, \theta, \delta, \gamma_{e'}, \sigma_{e'}$ to conclude $(\varsigma_{e'}, \theta(\delta(\gamma_{e'}(e')))) \in \mathcal{C}_{k-i_e}[\![\theta(t')]\!]$.

⟨2⟩10. Instantiate *this* with $j - i_e$ and σ_f to conclude $\langle \theta(\sigma_f + \sigma_{e'}), \theta(\delta(\gamma_{e'}(e'))) \rangle$ either takes $j - i_e$ steps to **err** or $\langle \sigma_f + \sigma'_f, e'_f \rangle$.

⟨2⟩11. CASE: $j - i_e$ steps to **err**
By OP_CONTEXT_ERR, the whole expression reduces to **err** in $j - i_e < k - i_e$ steps.

⟨2⟩12. CASE: $j - i_e$ steps to another heap-and-expression.
If it is not a value, then OP_CONTEXT runs $j - i_e$ times and we are done.

⟨2⟩13. If it is, then $\exists i_{e'} \leq j - i_e.\ (\varsigma'_f, v_{e'}) \in \mathcal{V}_{k-i_e-i'_e}[\![\theta(t')]\!]$ by 4.3.
So, OP_CONTEXT runs $i_{e'}$ times, and then we have the following.
Suffices: By 4.4 $i_{e'}$ times, $(\varsigma_f \star \varsigma'_f, e_f\, e'_f) \in \mathcal{C}_{k-i_e-i_{e'}}[\![\theta(t')]\!]$.

⟨2⟩14. Instantiate $(\varsigma_f, e_f) \in \mathcal{V}_{k-i_e}[\![\theta(t') \multimap \theta(t)]\!]$ with $k - i_e - i_{e'} \leq k - i_e$ and

$(\varsigma_{v'}, v_{e'}) \in \mathcal{V}_{k-i_e-i_{e'}}[\![\theta(t')]\!]$, to conclude $(\varsigma_f \star \varsigma_f', e_f \, e_f') \in \mathcal{C}_{k-i_e-i_{e'}}[\![\theta(t)]\!]$ as needed.

$\langle 1 \rangle 10.$ CASE: TY_GEN.

$\quad \langle 2 \rangle 1.$ By induction, $\forall k. \; _k[\![\Theta, fc; \Delta; \Gamma \vdash e : t]\!]$.

$\quad \langle 2 \rangle 2.$ LET: f be arbitrary; $\theta' \equiv \theta[fc \mapsto f]$.
$\qquad$ Instantiate induction hypothesis with $k-1, \theta', \delta, \gamma, \sigma$,
$\qquad$ to conclude $(\varsigma, \theta'(\gamma(\delta(e)))) \in \mathcal{C}_{k-1}[\![\theta'(t)]\!]$ (for all f, by 4.8).

$\quad \langle 2 \rangle 3.$ Instantiate *this* with j and $\emptyset$ to conclude $\langle \theta(\sigma), \theta'(\gamma(\delta(e))) \rangle$
$\qquad$ either takes j steps to **err** or a heap-and-expression $\langle \sigma', e' \rangle$ (for all f, by 4.8).

$\quad \langle 2 \rangle 4.$ CASE: j steps to **err**.
$\qquad$ By OP_CONTEXT_ERR, whole expression reduces to **err** in $j < k-1$ steps (for $f = fc$).

$\quad \langle 2 \rangle 5.$ CASE: j steps to another heap-and-expression.
$\qquad$ If it is not a value, then for $f = fc$, OP_CONTEXT runs j times and we are done.

$\quad \langle 2 \rangle 6.$ If it is, then $\exists i_e \le j. \; (\varsigma', e') \in \mathcal{V}_{k-1-i_e}[\![\theta'(t)]\!] \subseteq \mathcal{V}_{k-1-j}[\![\ldots]\!]$
$\qquad$ by 4.3 and 4.5 (for all f, by 4.8).

$\quad \langle 2 \rangle 7.$ So, OP_CONTEXT runs i_e times, and then we have the following.
$\qquad$ SUFFICES: By 4.4 i_e times, $(\varsigma', \mathbf{fun} \, fc \to e') \in \mathcal{V}_{k-i_e}[\![\theta(\forall fc. \, t)]\!]$ (for $f = fc$).

$\quad \langle 2 \rangle 8.$ ASSUME: Arbitrary f'.
$\qquad$ SUFFICES: $(\varsigma', e'[fc/f']) \in \mathcal{V}_{k-1-i_e}[\![\theta(t)[fc/f']]\!]$ (for $f = fc$).

$\quad \langle 2 \rangle 9.$ This is true by instantiating $\langle 2 \rangle 6$ with $f = f'$.

$\langle 1 \rangle 11.$ CASE: TY_SPC.

$\quad \langle 2 \rangle 1.$ By induction, $\forall k. \; _k[\![\Theta; \Delta; \Gamma \vdash e : \forall fc. \, t]\!]$.

$\quad \langle 2 \rangle 2.$ Instantiate with $k, \theta, \delta, \gamma, \sigma$ to conclude $(\varsigma, \theta(\delta(\gamma(e)))) \in \mathcal{C}_k[\![\theta(\forall fc. \, t)]\!]$.

$\quad \langle 2 \rangle 3.$ Instantiate *this* with j and $\emptyset$ and to conclude $\langle \theta(\sigma), \theta(\delta(\gamma(e))) \rangle$
$\qquad$ either takes j steps to **err** or a heap-and-expression $\langle \sigma_f, e_f \rangle$.

$\quad \langle 2 \rangle 4.$ CASE: j steps to **err**.
$\qquad$ By OP_CONTEXT_ERR, the whole expression reduces to **err** in $j < k$ steps.

$\quad \langle 2 \rangle 5.$ CASE: j steps to another heap-and-expression.
$\qquad$ If it is not a value, then OP_CONTEXT runs j times and we are done.

$\quad \langle 2 \rangle 6.$ If it is, then $\exists i_e \le j. \; (\varsigma_f, e_f) \in \mathcal{V}_{k-i_e}[\![\theta(\forall fc.t)]\!] \subseteq \mathcal{V}_{k-j}[\![\ldots]\!]$ by 4.3 and 4.5.
$\qquad$ So $e_f \equiv \mathbf{fun} \, fc \to v$ for some v.

$\quad \langle 2 \rangle 7.$ So, OP_CONTEXT runs i_e times, and then we have the following.
$\qquad$ SUFFICES: By 4.4 i_e times, $(\varsigma_f, (\mathbf{fun} \, fc \to v) \, [f]) \in \mathcal{C}_{k-i_e}[\![\theta(t[fc/f])]\!]$.
$\qquad$ SUFFICES: By 4.4 once more, $(\varsigma_f, v[fc/f]) \in \mathcal{C}_{k-i_e-1}[\![\theta(t[fc/f])]\!]$.

$\quad \langle 2 \rangle 8.$ This is true by instantiating $\langle 2 \rangle 6$ with f and 4.2.

$\langle 1 \rangle 12.$ CASE: TY_FIX.
$\quad$ SUFFICES: $(\emptyset, \theta(\delta(\mathbf{fix}(g, x : t, e : t')))) \in \mathcal{V}_k[\![\theta(t \multimap t')]\!]$, by 4.2 $(\sigma = \{\}, \gamma = [])$.
$\quad$ ASSUME: Arbitrary $j \le k$, $(\varsigma_v, v) \in \mathcal{V}_j[\![\theta(t)]\!]$ $(\varsigma = \emptyset,$ so $\varsigma \star \varsigma_v$ is defined$)$.

LET: $\tilde{e} \equiv \theta(\delta(e)))$.
SUFFICES: $(\varsigma_v, \mathbf{fix}(g, x : t, \tilde{e} : t') \; v) \in \mathcal{C}_j[\![\theta(t')]\!]$.
SUFFICES: $(\varsigma_v, \tilde{e}\,[x/v]\,[g/\mathbf{fix}(g, x : t, \tilde{e} : t')]) \in \mathcal{C}_{j-1}[\![\theta(t')]\!]$ by 4.4.

$\langle 2 \rangle 1$. By induction, $\forall k.\ {}_k[\![\Theta; \Delta, g : t \multimap t'; x : t \vdash e : t']\!]$.

$\langle 2 \rangle 2$. Instantiate this with $j - 1, \delta[g \mapsto \mathbf{fix}(g, x : t, \tilde{e} : t')], \gamma = [x \mapsto v], \sigma_v$.

$\langle 2 \rangle 3$. We have $(\emptyset, \mathbf{fix}(g, x : t, \tilde{e} : t')) \in \mathcal{V}_{j-1}[\![\theta(t \multimap t')]\!]$.

$\quad \langle 3 \rangle 1$. Again by induction (over k), $(\emptyset, \mathbf{fix}(g, x : t, \tilde{e} : t')) \in \mathcal{C}_{j-1}[\![\theta(t \multimap t')]\!]$.

$\quad \langle 3 \rangle 2$. Instantiate *this* with $j = 0$ and $\emptyset$ and we are done.

$\langle 2 \rangle 4$. We have $(\varsigma_v, v) \in \mathcal{V}_{j-1}[\![\theta(t)]\!]$ by assumption and 4.5.

$\langle 2 \rangle 5$. So we conclude $(\varsigma_v, \theta(\delta'(\gamma(e)))) \in \mathcal{C}_{j-1}[\![\theta(t')]\!]$ as required.

$\langle 1 \rangle 13$. CASE: TY_VAR_LIN.
PROVE: $(\varsigma, \theta(\delta(\gamma(x)))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

$\langle 2 \rangle 1$. $\Gamma = \{x : t\}$ by assumption of TY_VAR_LIN.

$\langle 2 \rangle 2$. SUFFICES: $(\varsigma, \gamma(x)) \in \mathcal{C}_k[\![\theta(t)]\!]$ by 3 (θ and δ irrelevant).

$\langle 2 \rangle 3$. By 2c, there exist $(\varsigma_x, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]$, such that $\varsigma = \varsigma_x$ and $\gamma = [x \mapsto v_x]$.

$\langle 2 \rangle 4$. Hence, $(\varsigma_x, v_x) \in \mathcal{C}_k[\![\theta(t)]\!]$, by 4.2.

$\langle 1 \rangle 14$. CASE: TY_VAR.
PROVE: $(\varsigma, \theta(\delta(\gamma(x)))) \in \mathcal{C}_k[\![\theta(t)]\!]$.

$\langle 2 \rangle 1$. $x : t \in \Delta$ and $\Gamma = \emptyset$ by assumption of TY_VAR.

$\langle 2 \rangle 2$. SUFFICES: $(\emptyset, \delta(x)) \in \mathcal{C}_k[\![\theta(t)]\!]$ by 3.

$\langle 2 \rangle 3$. By 2b, there exists v_x such that $(\emptyset, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]$ (θ irrelevant and γ empty).

$\langle 2 \rangle 4$. Hence, $(\emptyset, v_x) \in \mathcal{C}_k[\![\theta(t)]\!]$, by 4.2.

$\langle 1 \rangle 15$. CASE: TY_UNIT_INTRO.
True by 4.2 and definition of $\mathcal{V}_k[\![\mathbf{unit}]\!]$.

$\langle 1 \rangle 16$. CASE: TY_BOOL_TRUE, TY_BOOL_FALSE, TY_INT_INTRO, TY_ELT_INTRO.
Similar to TY_UNIT_INTRO.

6 Additional Details

6.1 Well-formed types

$\boxed{\Theta \vdash f \, \mathsf{Perm}}$ Well-formed fractional permissions

$$\frac{fc \in \Theta}{\Theta \vdash fc \, \mathsf{Perm}} \quad \text{WF\_Perm\_Var}$$

$$\frac{}{\Theta \vdash 1 \, \mathsf{Perm}} \quad \text{WF\_Perm\_Zero}$$

$$\frac{\Theta \vdash f \, \mathsf{Perm}}{\Theta \vdash \frac{1}{2}f \, \mathsf{Perm}} \quad \text{WF\_Perm\_Succ}$$

$\boxed{\Theta \vdash t \, \mathsf{Type}}$ Well-formed types

$$\frac{}{\Theta \vdash \mathbf{unit} \, \mathsf{Type}} \quad \text{WF\_Type\_Unit}$$

$$\frac{}{\Theta \vdash \mathbf{bool} \, \mathsf{Type}} \quad \text{WF\_Type\_Bool}$$

$$\frac{}{\Theta \vdash \mathbf{int} \, \mathsf{Type}} \quad \text{WF\_Type\_Int}$$

$$\frac{}{\Theta \vdash \mathbf{elt} \, \mathsf{Type}} \quad \text{WF\_Type\_Elt}$$

$$\frac{\Theta \vdash f \, \mathsf{Perm}}{\Theta \vdash f \, \mathbf{arr} \, \mathsf{Type}} \quad \text{WF\_Type\_Array}$$

$$\frac{\Theta \vdash t \, \mathsf{Type}}{\Theta \vdash \,!t \, \mathsf{Type}} \quad \text{WF\_Type\_Bang}$$

$$\frac{\Theta, fc \vdash t \, \mathsf{Type}}{\Theta \vdash \forall fc.t \, \mathsf{Type}} \quad \text{WF\_Type\_Gen}$$

$$\frac{\Theta \vdash t \, \mathsf{Type} \quad \Theta \vdash t' \, \mathsf{Type}}{\Theta \vdash t \otimes t' \, \mathsf{Type}} \quad \text{WF\_Type\_Pair}$$

$$\frac{\Theta \vdash t \, \mathsf{Type} \quad \Theta \vdash t' \, \mathsf{Type}}{\Theta \vdash t \multimap t' \, \mathsf{Type}} \quad \text{WF\_Type\_Lolly}$$

6.2 Grammar Definition

$$
\begin{array}{llll}
m & ::= & & \text{matrix expressions} \\
  & | & M & \text{matrix variables} \\
  & | & m + m' & \text{matrix addition} \\
  & | & m \, m' & \text{matrix multiplication} \\
  & | & (m) \quad \mathsf{S} & \\
\end{array}
$$

f	::=			fractional permission
	|	fc		variable
	|	1		whole permission
	|	$\frac{1}{2}f$		

t	::=			linear type
	|	**unit**		unit
	|	**bool**		boolean (true/false)
	|	**int**		63-bit integers
	|	**elt**		array element
	|	f **arr**		arrays
	|	f **mat**		matrices
	|	$!t$		multiple-use type
	|	$\forall fc.t$	bind fc in t	frac. perm. generalisation
	|	$t \otimes t'$		pair
	|	$t \multimap t'$		linear function
	|	(t)	S	parentheses

p	::=		primitive
	|	**not**	boolean negation
	|	$(+)$	integer addition
	|	$(-)$	integer subtraction
	|	$(*)$	integer multiplication
	|	$(/)$	integer division
	|	$(=)$	integer equality
	|	$(<)$	integer less-than
	|	$(+.)$	element addition
	|	$(-.)$	element subtraction
	|	$(*.)$	element multiplication
	|	$(/.)$	element division
	|	$(= .)$	element equality
	|	$(< .)$	element less-than
	|	**set**	array index assignment
	|	**get**	array indexing
	|	**share**	share array
	|	**unshare**	unshare array
	|	**free**	free arrary
	|	**array**	Owl: make array
	|	**copy**	Owl: copy array
	|	**sin**	Owl: map sine over array
	|	**hypot**	Owl: $x_i := \sqrt{x_i^2 + y_i^2}$
	|	**asum**	BLAS: $\sum_i |x_i|$
	|	**axpy**	BLAS: $x := \alpha x + y$
	|	**dot**	BLAS: $x \cdot y$
	|	**rotmg**	BLAS: see its docs
	|	**scal**	BLAS: $x := \alpha x$
	|	**amax**	BLAS: $\operatorname{argmax} i : x_i$
	|	**setM**	matrix index assignment

	getM		matrix indexing
	shareM		share matrix
	unshareM		unshare matrix
	freeM		free matrix
	matrix		Owl: make matrix
	copyM		Owl: copy matrix
	copyM_to		Owl: copy matrix onto another
	sizeM		dimension of matrix
	trnsp		transpose matrix
	gemm		BLAS: $C := \alpha A^{T?} B^{T?} + \beta C$
	symm		BLAS: $C := \alpha AB + \beta C$
	posv		BLAS: Cholesky decomp. and solve
	potrs		BLAS: solve with given Cholesky
	syrk		BLAS: $C := \alpha A^{T?} A^{T?} + \beta C$

v ::=   values

	p		primitives
	x		variable
	$()$		unit introduction
	true		true
	false		false
	k		integer
	l		heap location
	el		array element
	Many v		!-introduction
	fun $fc \to v$		frac. perm. abstraction
	(v, v')		pair introduction
	fun $x : t \to e$	bind x in e	abstraction
	fix $(g, x : t, e : t')$	bind $g \cup x$ in e	fixpoint
	(v)	S	parentheses

e ::=   expression

	p		primitives
	x		variable
	let $x = e$ **in** e'	bind x in e'	let binding
	$()$		unit introduction
	let $() = e$ **in** e'		unit elimination
	true		true
	false		false
	if e **then** e_1 **else** e_2		if
	k		integer
	l		heap location
	el		array element
	Many e		!-introduction
	let Many $x = e$ **in** e'		!-elimination
	fun $fc \to e$		frac. perm. abstraction
	$e[f]$		frac. perm. specialisation
	(e, e')		pair introduction

	let $(a, b) = e$ **in** e'	bind $a \cup b$ in e'	pair elimination
	fun $x : t \to e$	bind x in e	abstraction
	$e\ e'$		application
	fix $(g, x : t, e : t')$	bind $g \cup x$ in e	fixpoint
	(e)	S	parentheses

C ::=   evaluation contexts

	let $x = [-]$ **in** e	bind x in e	let binding
	let $() = [-]$ **in** e		unit elimination
	if $[-]$ **then** e_1 **else** e_2		if
	Many $[-]$		!-introduction
	let Many $x = [-]$ **in** e		!-elimination
	fun $fc \to [-]$		frac. perm. abstraction
	$[-][f]$		frac. perm. specialisation
	$([-], e)$		pair introduction
	$(v, [-])$		pair introduction
	let $(a, b) = [-]$ **in** e	bind $a \cup b$ in e	pair elimination
	$[-]e$		application
	$v[-]$		application

Θ ::=   fractional permission environment

| | $\cdot$ | |
| | Θ, fc | |

Γ ::=   linear types environment

	$\cdot$	
	$\Gamma, x : t$	
	Γ, Γ'	

Δ ::=   intuitionistic types environment

| | $\cdot$ | |
| | $\Delta, x : t$ | |

σ ::=   heap (multiset of triples)

| | $\{\}$ | empty heap |
| | $\sigma + \{l \mapsto_f m_{k_1, k_2}\}$ | location l points to matrix m |

Config ::=   result of small step

| | $\langle \sigma, e \rangle$ | heap and expression |
| | **err** | error |